## 按照开发阶段划分

1）单元测试--针对系统的一个小的单元模块进行测试的

测试阶段：编码后，编码前（TDD:Test-Driven-Development测试驱动开发）先写测试用例，跑测试用例，开发人员根据测试用例产生的异常区补充开发代码

测试对象：最小模块

测试人员：白盒测试工程师或开发工程师

测试依据：代码和注释+详细设计文档

测试方法：白盒测试

测试内容：模块接口测试（我们单元测试测试每个单元是借助接口来调用的，给接口一个相应的输入和输出，输入参数，参数个数以及参数的类型，参数的顺序要符合我们的接口文档），局部数据结构测试，路径测试（if else  switch），错误处理测试（try catch finally），边界测试（for，while）

单元测试框架：java的单元测试的框架  Junit

2）集成测试：A B C D四个模块  A if B else C D

测试阶段：单元测试之后

测试对象：模块间的接口

测试人员：白盒测试工程师或开发工程师

测试依据：单元测试模块+概要设计文档

测试方法：黑盒测试与白盒测试相结合

测试内容：模块之间数据传输（数据在不同的单元逻辑之间流转）、模块之间功能的冲突（任何一个模块，他的功能都不会影响另一个模块）、模块功能的正确性、全局数据结构、单个模块的缺陷对整个系统的影响

3）系统测试  （全面测试）

主要做黑盒测试

测试阶段：开发完成后，集成测试完以后

测试人员：黑盒测试工程师

测试内容：界面 功能 性能 易用性 安全性 兼容性等

局部数据结构属于下面哪一个测试阶段？

A、集成测试  B、系统测试  C、单元测试  D、验收测试

答案：C

（1）回归测试：当系统的代码进行了修改的时候，为了防止新添加的代码对系统引入新的错误而进行的测试（添加新需求 修改BUG）  自动化测试

（2）冒烟测试：准入原则   对系统的核心功能和主要流程进行的测试

    决定我们测试人员是否接受系统进行正式测试的依据

    针对冒烟测试，提出提测用例

4）验收测试

测试阶段：系统测试之后，用户进行测试

测试内容：界面 功能 性能 易用性 安全性 兼容性等 文档 功能说明文档 开发文档 使用说明书

## 2、按照测试的实施组织：

2.1 α测试：把用户（除了开发和测试以外，公司其他的人）请到开发现场进行调试；

            及时解决发现问题（优点）

            环境受整个开发环境的限制（缺点）

2.2 β测试：用户在正常的使用环境进行测试，通常一个周期测试完成，把问题整理成文档，反馈给开发人员

            测试时间比较分散（缺点）

            用户在真实的使用环境下进行测试（优点）

            β测试在α测试之后

2.3 第三方

第三方软件测评机构（根据行业里面的标准和规范进行测试的）

3、按是否运行划分（代码运行  动态测试）

3.1 静态测试：看代码的人要比写代码的人要求更高

3.2 动态测试

## 4、按是否手工划分

4.1手工测试

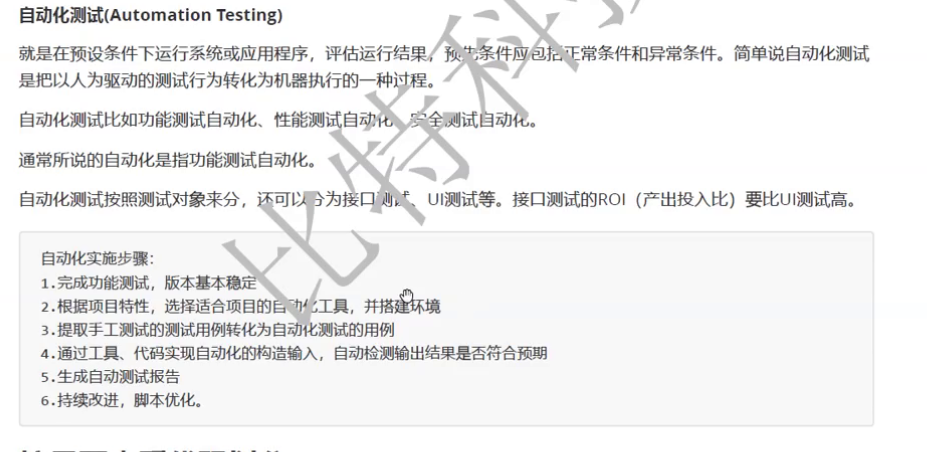
如果进行大量的测试容易出错（不可控的因素）

比较灵活，可以进行发散性的测试

4.2 自动化测试

前提：在系统功能比较稳定的前提下才可以做自动化测试

自动化测试的最大的价值：脚本的重复利用率，重复利用率越高价值越大



## 1、按照是否查看代码划分

**黑盒测试：**不关心软件内部的逻辑，结构，只关心输入输出是否达到我们的预期，相当于把测试的软件看成一个只有输入和输出的黑色盒子

黑盒测试设计测试用例的方法：

等价类、边界值、因果图法、正交法、场景法、错误猜测法

白盒测试：研究软件内部的程序逻辑和结构，验证是否满足软件需求，相当于把软件当成一个能看见内部结构的白色的盒子去测试

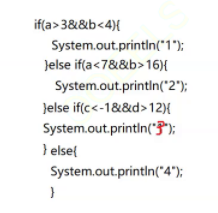
**白盒测试方法：**判定覆盖，条件覆盖，判定和条件的组合，判定组合，条件组合，语句覆盖，逻辑覆盖，路径覆盖

语句覆盖：最弱的，不考虑逻辑性，就是看看代码是不是能运行

逻辑覆盖：创造异常，看看能不能把异常catch住，如果没有异常，看看是否执行finally，还有switch，case

路径覆盖：

判定覆盖：把每一个判定都要走过去，看看是否是按照写的程序进行下去，让判定条件都走一遍



**灰盒测试：**介于白盒测试和黑盒测试之间，既关心输入和输出，有关心程序内部结构，按是否查看代码去划分

**单元测试：**给电脑装插件 File -- Settings -- Plugins -- Junit

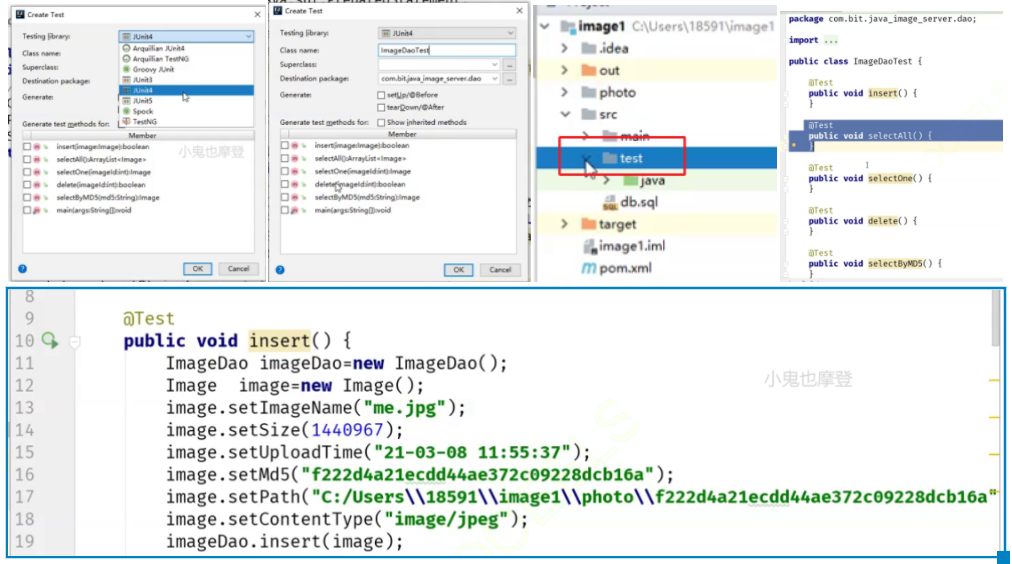
1）安装插件：File -- Settings -- Plugins -- Junit

2）找到要进行单元测试的类，选中类名，执行Ctrl+shift+T。创建单元测试类

3）写单元测试方法（生成在）

单元测试要在项目上做

选中类方法，Ctrl+shift+T出现，按回车



测试里面的数据可以直接从数据库中拿出来。比如insert方法，插入，插入之前有8条数据，插入之后，有9条数据，那么插入成功

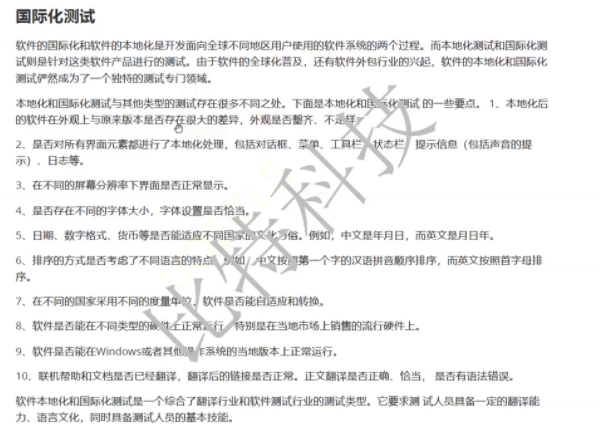
通过代码测试，叫做单元测试，这个方法也算是白盒测试

## 2、按测试的地域去划分

国际化测试：

什么是软件国际化：在设计软件的时候，采用的一种工程技术，使得软件在转换不同国家的语言，风俗习惯的时候，不需要修改源码的技术，就叫做软件国际化。Word，Windows系统，苹果手机在中国都是默认是中文的，或者可以切换语言

比如某某人去世，搜索到的是灰白色的



本地测试

## 3、按测试对象划分

测试用例

1）业务测试：过程

场景法：考虑正常异常的一些流程

了解业务

2）界面测试：

文字大小、字体、排版、颜色、粗细

图片清晰度、大小、排版、是否重叠、色彩

页面整体排版 信息提示弹出框

控件  对话框 查询框 滚动条 弹出框

按钮的亮度显示  有效的按钮高亮显示 无效的按钮置灰

响应式测试:页面可以随着屏幕大小的不同而变换

响应式页面的测试：文字可以在不同分辨率下能够完成并且清洗战士；图片在不同分辨率下能够正确排版，清晰展示，不重叠；界面的功能在不同的分辨率下都可以正常的使用；不同的界面在不同分辨率下切换的时候是平滑的过度；在不同的分辨率下，界面的展示要严格的按照UI设计稿来进行展示

3）功能测试：

4）兼容性：

5）安全性：

3、容错性测试

当系统出现一些异常，活着用户输入错误的信息，进行异常的操作，系统可以自我处理这些错误的情况，不会出现页面的崩溃，页面的异常的情况，可以给用户一个很好的提示体验

1）数据容错性：时间、日期、数字  人民币：元角分 小数点后两位   注册密码  9-16位

2）校验容错性：查询信息前后加空格 自动去掉 再进行查询；大小写转换；验证码；数据信息一致性；个人信息的填写：身份证号或密码前后至少两次；信息界别，填写表单的时候，异常关闭（网络、没电、人力）是否会自动保存；同一个数据被不同人操作时，是否会有锁定；同一个数据，同一个信息，在不同的平台上（同一个账号下）是不是信息会同步的

3）界面容错性：复杂的操作，有说明提示；用户在执行有风险的操作的时候，会给出提示

4）环境容错性：网络、电源、服务器  无缝切换备用的网络、电源或者服务器就能够保证用户的无感知

5）灾难恢复性测试：认为的让服务器或者服务器所依赖的环境法何时能鼓掌，测试系统的自我修复能力；系统碰到这些极端情况的时候，系统回复的功能和里面的信息所需要的时间用户是否能够感知；系统碰到这些极端情况的时候，恢复系统的功能和里面的信息所需要的时间用户能否接受；

4、文档测试

完整性，正确性，一致性，术语是否专业，易用性

5、兼容性测试：

1）系通所在平台的兼容性：比如要测web系统，首先各种操作系统 Windows iOS 不同的操作系统要进行测试，不同操作系统的不同版本也要进行测试，浏览器要装在不同的操作系统上；不同的浏览器，网页是否可以正常展示和使用 Chrome Firefox edge IE 360 搜狗 QQ；不同浏览器的市场上主流的版本，不同浏览器的内核不一样；APP 不同的机型，不同机型的不同的操作系统  iOS Android 不同从做系统的主流版本上进行测试 ipone 华为 OPPO vivo 魅族 小米

兼容性测试在不同平台上测试的功能都是一样的，所以可以用自动化测试

2）软件本身向前或者向后的兼容性：

3）软件和其他相关的软件的兼容性：淘宝 支付宝 盒马鲜生 1688

4）数据的兼容性：同一个软件数据兼容性

6、易用性测试

遵循一定的 易用性 标准来开发软件 前期积累经验 越来越复合人的使用习惯和使用需要

直观性 软件的展示要和软件定位相关

7、安装测试（卸载）

APP：应用商店 网页 安装包 命令行

系统：官网

8、安全性测试

病毒 SQL的注入 xss OS攻击 木马



APP基本功能的测试

9、性能的测试

能够快速的响应用户的请求（3S/5S/8S 358原则）

系统能够负载所需要的用户数量

能够处理系统所需要的事务数量

在负载和事务处理的高峰期，系统可以稳定

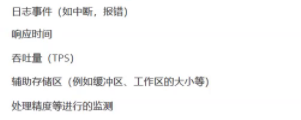
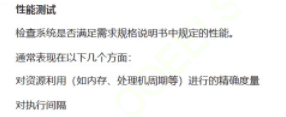
在系统能够处理的最高用户负载和事务数量的时候，用户可以获得良好的体验

性能测试哪些方面？

对资源的利用率

响应时间：用户发送请求到用户所期待的响应（页面、信息）展示出来；

吞吐量：系统在单位时间处理的信息量  字节 事务处理量  http请求数



日志的级别 ： info debug warn error（自左而右，级别越来越高，级别越低越详细）